NTopo: Mesh-free Topology Optimization using Implicit Neural Representations

— Supplementary Material —

Jonas Zehnder
Department of Computer Science
and Operations Research
Université de Montréal
jonas.zehnder@umontreal.ca

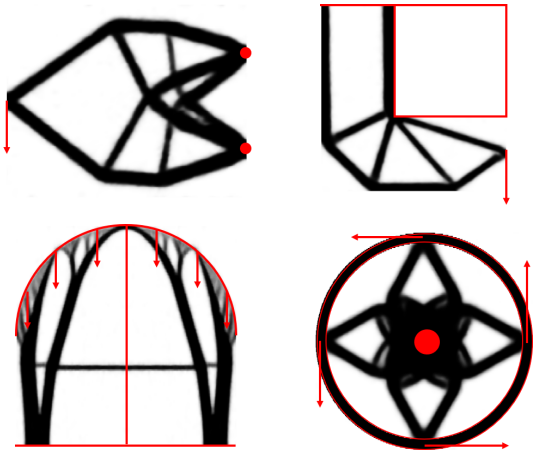
Yue Li
Department of Computer Science
ETH Zurich
yue.li@inf.ethz.ch

Stelian Coros
Department of Computer Science
ETH Zurich
scoros@inf.ethz.ch

Bernhard Thomaszewski
Department of Computer Science
ETH Zurich
bthomasz@ethz.ch

1 Additional Results Obtained with our Method

We show more results obtained using our mesh-free topology optimization approach.



Top-left: Two point locations are fixed on the right with a point-wise force applied on the left. Top-right The structure is held on the top and a point-wise force is applied on the bottom right. The top right of the density field is constrained to be empty. Bottom-left: The structure is held on the bottom and a distributed force field is applied across a half-circle. Bottom-right: The structure is held in the middle and four tangential forces are applied on an outer annulus of material. The target volume ratios in the same order are 30%, 20%, 25% and 40%.

35th Conference on Neural Information Processing Systems (NeurIPS 2021).

2 Sensitivity Analysis.

Here we provide the derivation of the total gradient $d\mathcal{L}_{\text{comp}}/d\theta$ for the density network weights θ, taking into account that when we change θ the displacement u changes, such that it stays in static equilibrium.

First, we would like to note that there are two different perspectives on the optimization problem (1) in the main section of the paper. The difference in formulations is due to the choice of

1. eliminating constraints and keeping just one set of variables (ρ), in this case $u = Simulation(\rho)$
2. using separate variables, ρ and u, and introducing explicit equilibrium constraints $\nabla\mathcal{L}_{\text{sim}} = 0$

Formulation 1) is used in the main paper, as it is in our opinion more easily readable. Formulation 2) corresponds to the standard formulation of constrained optimization problems using Lagrange multipliers. In the Lagrangian L, all arguments (u, ρ and the Lagrange multipliers) are independent and are only coupled through the solution condition $\nabla L = 0$. Here, the derivation of the gradient will rely on formulation 2).

One way of computing the total gradient of the compliance objective is to solve for the discrete adjoint variables. Solving the discretized adjoint problem takes the form of

$$\frac{\partial^2 \mathcal{L}_{\text{sim}}}{\partial \gamma^2}\lambda = -\frac{\partial \mathcal{L}_{\text{comp}}}{\partial \gamma} \tag{1}$$

however this is a very large dense linear system which would be costly to solve, instead we rely on a point-wise argument to derive the adjoint gradient. We apply the chain rule with the neural network to get the total gradient

$$\frac{d\mathcal{L}_{\text{comp}}}{d\theta} = \int_\Omega \frac{de(\omega)}{d\rho(\omega)} \frac{\partial \rho(\omega)}{\partial \theta}\, d\omega \tag{2}$$

where $de(\omega)/d\rho(\omega)$ is the point-wise total gradient of the point-wise compliance e. To derive this point-wise gradient we start by introducing the Lagrangian

$$\begin{aligned} L(u,\rho,\lambda,\mu) &= \int_\Omega e + \lambda^T(\nabla\cdot\sigma(u) - f)\, d\omega + \int_{\partial\Omega} \mu^T(\sigma(u)n)\, d\omega && (3)\\ &= \int_\Omega e - \nabla\lambda : \sigma(u)\, d\omega - \lambda^T f\, d\omega + \int_{\partial\Omega} -\lambda^T(\sigma(u)n) + \mu^T(\sigma(u)n)\, d\omega && (4)\\ &= \int_\Omega e - \epsilon(\lambda) : C : \epsilon(u) - \lambda^T f\, d\omega + \int_{\partial\Omega} (\mu-\lambda)^T(\sigma(u)n)\, d\omega && (5) \end{aligned}$$

where we used integration by parts and leveraged the fact that $\nabla\lambda : \sigma = \epsilon(\lambda) : \sigma$ due to the symmetry of the stress tensor σ. C is the stiffness tensor in Hooke's law, that is, $\sigma = C : \epsilon$ and $\sigma_{ij} = \sum_{kl} C_{ijkl}\epsilon_{kl}$.

In the case of topology optimization for linear elasticity the adjoint variables are readily available through $\lambda = \mu = u$ [1, 2]. Here we briefly show the derivation:

We denote the directional derivative of F w.r.t. x in the direction of h as

$$D_{x,h}F(x) = \lim_{t\to 0} \frac{F(x+th) - F(x)}{t} \tag{6}$$

with which the adjoint variables are defined by [3]

$$\forall h \in (H^1)^2 : \quad D_{u,h}L(u,\rho,\lambda,\mu) = 0\,. \tag{7}$$

and by applying the definition of the directional derivative, this leads to

$$\forall h \in (H^1)^2 : \quad D_{u,h}L(u,\rho,\lambda,\mu) = \int \epsilon(h) : C : \epsilon(u) - \epsilon(h) : C : \epsilon(\lambda)\, d\omega = 0 \tag{8}$$

which is true for $\lambda = u$.

We can then plugin the adjoint variables into the Lagrangian to get the point-wise gradient of the compliance with respect to the density as

$$\forall \delta\rho \in H^1 : D_{\rho,\delta\rho} L(u, \rho, \lambda, \mu) = \int_\Omega -\delta\rho \, \frac{1}{2}\epsilon(u) : \frac{\partial C}{\partial \rho} : \epsilon(u) \, d\omega \tag{9}$$

Since this holds for all functions $\delta\rho$ in H^1 we can conclude that the point-wise gradient is

$$\frac{de(\omega)}{d\rho(\omega)} := -\frac{1}{2}\epsilon(\omega) : \frac{\partial C(\omega)}{\partial \rho(\omega)} : \epsilon(\omega) = -\frac{\partial e(\omega)}{\partial \rho(\omega)} \tag{10}$$

which notably has the opposite sign of the partial gradient $\partial e / \partial \rho$ where the implicit change in the displacement u is not taken into account.

Batch density-space gradient By sampling a batch of ω^b and $\rho^b = \rho(\omega^b)$ we can approximate the integrals and get a discrete version of the loss

$$\mathcal{L}_{\text{comp}} = \int_\Omega e \, d\omega \approx \mathcal{L}'_{\text{comp}} = \frac{|\Omega|}{n} \sum_i^n e(\omega_i^b) \tag{11}$$

using the derivations above we arrive at

$$\frac{d\mathcal{L}_{\text{comp}}}{d\theta} = \int_\Omega \frac{de(\omega)}{d\rho(\omega)} \frac{\partial \rho(\omega)}{\partial \theta} \, d\omega = -\int_\Omega \frac{\partial e(\omega)}{\partial \rho(\omega)} \frac{\partial \rho(\omega)}{\partial \theta} \, d\omega \tag{12}$$

$$\approx -\frac{|\Omega|}{n} \sum_i^n \frac{\partial e(\omega_i^b)}{\partial \rho(\omega_i^b)} \frac{\partial \rho(\omega_i^b)}{\partial \theta} = -\frac{\partial \mathcal{L}'_{\text{comp}}}{\partial \rho^b} \frac{\partial \rho^b}{\partial \theta} \tag{13}$$

$$= \frac{d\mathcal{L}'_{\text{comp}}}{d\rho^b} \frac{\partial \rho^b}{\partial \theta}. \tag{14}$$

3 Additional Information for the Results

Here we provide more detailed descriptions of the results in the main paper. See the following table for the domain sizes and boundary condition enforcements.

	Ω	$\hat{V}$	$d_x(\omega)$	$d_y(\omega)$	$d_z(\omega)$
Short Beam	1.5×0.5	0.5	ω_x	ω_x	—
Distributed	1.5×0.5	0.4	ω_x	ω_x	—
Long Beam	1.0×0.5	0.4	$\omega_x(\omega_x - 1.0)$	ω_x	—
Bridge	1.0×0.5	0.4	$\omega_x(\omega_x - 1.0)$	ω_x	—
3D Beam	$1.0 \times 0.5 \times 0.25$	0.2	ω_x	ω_x	$\omega_x(\omega_z - 0.25)$
3D Bridge	$1.0 \times 0.5 \times 0.25$	0.2	$\omega_x(\omega_x - 1.0)$	ω_x	$\omega_x(\omega_z - 0.25)$

We apply a concentrated force at the bottom right corner for the *Short Beam* example and distributed forces on the top boundary for the *Distributed* example, all along the negative y-axis. A concentrated force is applied at the bottom right corner of the new design domain for the *Long Beam example* and distributed loading forces are added to the bottom boundary for the *Bridge* example.

The design domains for both the *Long Beam* and the *Bridge* are 2.0×0.5, however, due to the symmetric configuration, we only perform simulation and optimization on half of the domain to avoid unnecessary computations. The domains are then 1.0×0.5. For visualization, we mirror the other half. We use the term $\omega_x(\omega_x - 1)$ in $d_x(\omega)$, this enforces the left boundary to have no displacement in x direction and the normal displacements of the right boundary to vanish, similar constructions are used in the other examples.

For the $3D\ Beam$ example, we run our method only on half of the design domain along z-axis, due to symmetry and the actual design domain is $1.0 \times 0.5 \times 0.25$. Distributed forces are applied at $x = 1.0,\ y = 0$ and $z \in [0.0, 0.25]$.

For the $3D\ Bridge$ example our method is only performed on a quarter of the design domain $2.0 \times 0.5 \times 0.5$, where both x and z dimension are reduced to half. Forces are applied to the bottom plane where $y = 0,\ x \in [0.0, 1.0]$ and $z \in [0.0, 0.25]$. All forces applied are along the negative y-axis.

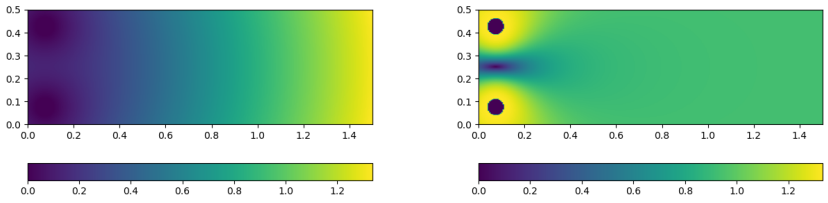


Figure 1: Left: Length factor $d(\omega)$ of the curved boundary example. Right: Norm of gradient of length factor, $\|\partial d/\partial \omega\|$

3.1 Curved Boundaries

We have used two additional types of constraints in these examples, constraining densities and constraining displacements on curved boundary. Since derivatives of the density field $\rho(\omega)$ do no appear in the objectives directly, constraining densities is rather straight forward. We just overwrite the density when the point ω falls into a constraint region Ω^c_ρ:

$$\tilde{\rho}(\omega) = \begin{cases} \rho(\omega) & \text{if } \omega \notin \Omega^c_\rho \\ \rho^c(\omega) & \text{if } \omega \in \Omega^c_\rho \end{cases} \tag{15}$$

and use the resulting field $\tilde{\rho}$ in place of ρ. For the displacement u, applying constraints is more difficult since its derivatives appear in the objectives, requiring that the field u is sufficiently smooth inside the domain. We use a length factor d [4], to define the constrained displacement field

$$u(\omega) = d(\omega)\Phi_u(\omega) \tag{16}$$

where d has to be zero on the boundary and have non-zero first-order derivatives on the boundary (otherwise the first-order derivatives are also constrained to zero on the boundary, which we do not want). Here we show a simple analytic method for defining the length factor and is accurate on the whole primitives not only on sampled points: we allow for n primitives that require the ability to compute a C^1 continuous distance function, giving $d_1^2, \dots d_n^2$ squared distances. We then combine them to construct a smooth length factor

$$d(\omega) = \sqrt{\mathrm{mix}(d_1^2, \dots, d_n^2)} \tag{17}$$

where mix repeatedly merges two distances by applying a function $m(\cdot, \cdot)$ in a tree like fashion, where we use the following function

$$m(d_1^2, d_2^2) = \frac{d_1^2 d_2^2}{d_1^2 + d_2^2 + \varepsilon} \qquad \varepsilon = 10^{-4} \tag{18}$$

This function is basically the power smooth min [5] with $k = 2$ plus ε, notably this function is zero when either distance d_1 or distance d_2 is zero and seems to have nice first-order derivative properties.

Data of curved boundary examples The domain for both examples is 1.5×0.5.

Three-hole-design: The bottom left of the domain is $[0, 0]$. The holes have inner radius 0.035 and outer radius 0.075, they are located at {[0.075, 0.425], [0.075, 0.075], [1.425, 0.075]}. The force is applied at $[1.425, 0.04]$. The length factor of one circle was defined using

$$d(\omega) = \begin{cases} 0 & \text{if } \|c - \omega\| < r \\ \|c - \omega\| - r & \text{otherwise} \end{cases} \tag{19}$$

where c is the center and r is the radius, and then the length factors of two circles were combined using the procedure explained above.

Hole in the middle-design: The radius of the circle is 0.2 and the center of the circle is at the center of the domain.

References

[1] Grégoire Allaire, François Jouve, and Anca-Maria Toader. Structural optimization using sensitivity analysis and a level-set method. *Journal of computational physics*, 194(1):363–393, 2004.

[2] Zhen Luo, Liping Chen, Jingzhou Yang, Y Zhang, and K Abdel-Malek. Compliant mechanism design using multi-objective topology optimization scheme of continuum structures. *Structural and Multidisciplinary Optimization*, 30(2):142–154, 2005.

[3] Fredi Tröltzsch. *Optimal control of partial differential equations: theory, methods, and applications*, volume 112. American Mathematical Soc., 2010.

[4] Kevin Stanley McFall and James Robert Mahan. Artificial neural network method for solution of boundary value problems with exact satisfaction of arbitrary boundary conditions. *IEEE Transactions on Neural Networks*, 20(8):1221–1233, 2009.

[5] Inigo Quilez. smooth minimum. `https://www.iquilezles.org/www/articles/smin/smin.htm`. Accessed: 2021-15-01.